Penerapan Algoritma A* Untuk Pencarian Rute Terdekat Pada Permainan Berbasis Ubin (Tile Based Game)

by Febriana Santi Wahyuni

Submission date: 23-Sep-2019 09:58AM (UTC+0700)
Submission ID: 1177900842
File name: febriana_seniati_2016.pdf (348.9K)
Word count: 2095
Character count: 12838

SEMINAR NASIONAL INOVASI DAN APLIKASI
TEKNOLOGI DI INDUSTRI (SENIATI) 2016
ISSN : 2085-4218

Penerapan Algoritma A* Untuk Pencarian Rute Terdekat Pada Permainan Berbasis Ubin *(Tile Based Game)*

Febriana Santi Wahyuni[1,*], Sandy Nataly Mantja[1]
1 T.Informatika Fakultas Teknologi Industri Institut Teknologi Nasional Malang
E-mail : vbryana@yahoo.com

Abstrak. Penggunaan Kecerdasan Buatan *(Artificial Intelligence)* pada sebuah permainan bertujuan untuk membuatnya lebih menarik. Beberapa jenis kecerdasan buatan dapat diterapkan pada sebuah game, salah satunya yaitu algoritma *pathfinding* yang merupakan kecerdasan buatan yang dapat menemukan lintasan terpendek. Salah satu algoritma *pathfinding* yang seringkali digunakan adalah Algoritma A*. Algoritma A*(AStar) merupakan algoritma *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best-First Search*. *Cost* yang diperhitungkan didapat dari *cost* sebenarnya ditambah dengan *cost* perkiraan. Dengan perhitungan *cost* seperti ini, algoritma A*(AStar) dapat optimal dan lengkap dalam menentukan jalur. Permainan berbasis ubin merupakan permainan yang menggunakan ubin *(tile)* sebagai salah satu elemen dasar dalam permainan. Dimana gambar-gambar *background game* yang terlalu besar dipotong menjadi kotak-kotak kecil seperti ubin atau dikenal dengan istilah *tiles*. Selanjutnya potongan-potongan map tersebut akan dipanggil ke dalam *game* dengan fungsi array untuk membentuk pola *map game*. Berdasarkan hasil pengujian algoritma A*(AStar) yang digunakan pada *game* ini teruji 100% efektif untuk menemukan jalur terpendek dari pergerakan objek dalam *game*. Pengujian heuristik pada algoritma A*(AStar), jalur terpendek yang ditemukan adalah sama, namun rute yang di gunakan dari tiap heuristik berbeda

Kata Kunci: ***Artificial Intelligent*, *Pathfinding*, Algoritma A*, *Tile Based Game***

1. Pendahuluan

Perkembangan permainan sampai saat ini sangat pesat, di tandai dengan munculnya jenis-jenis permainan yang baru. Tetapi hal tersebut tidak membuat jenis-jenis permainan yang sederhana seperti jenis permainan berbasis ubin tidak berkembang. *Permainan berbasis ubin* merupakan permainan yang menggunakan ubin *(tile)* sebagai salah satu elemen dasar dalam permainan. Permainan jenis ini sudah lama sekali digunakan dalam pembuatan *permainan*. Sejak komputer belum memiliki spesifikasi yang canggih seperti saat ini. Kecepatan yang lambat dan kapasitas memori yang terbatas membuat para pembuat *game* harus memutar otak untuk menemukan cara cerdas dalam membuat *game* yang terlihat bagus dan berjalan lebih cepat. Untuk itulah digunakan *Tile-Based* sebagai dasar dalam pembuatan *game*, dimana gambar-gambar *background game* yang terlalu besar dan dapat memperlambat jalannya permainan dalam *game* dipotong menjadi kotak-kotak kecil seperti ubin atau dikenal dengan istilah *tiles*. Nantinya potongan-potongan map tersebut akan dipanggil ke dalam *game* dengan fungsi array untuk membentuk pola *map game*.

Penggunaan Kecerdasan Buatan *(Artificial Intelligence)* pada sebuah permainan bertujuan untuk membuatnya lebih menarik. Beberapa jenis kecerdasan buatan dapat diterapkan pada sebuah game, salah satunya yaitu algoritma *pathfinding* yang merupakan kecerdasan buatan yang dapat menemukan lintasan terpendek. Salah satu algoritma *pathfinding* yang seringkali digunakan adalah Algoritma A*. Algoritma A*(AStar) merupakan algoritma *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best-First Search*. *Cost* yang diperhitungkan didapat dari *cost* sebenarnya ditambah dengan *cost* perkiraan. Dengan perhitungan *cost* seperti ini, algoritma A*(AStar) adalah *complete* dan optimal dalam menentukan jalur.

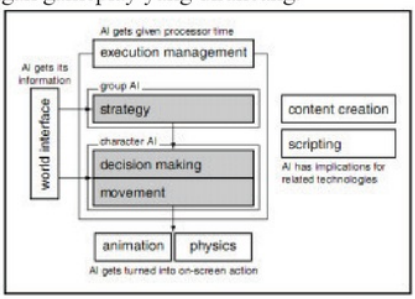
Tujuan

Adapun tujuan dari penelitian ini adalah menerapkan Algoritma A* untuk menemukan lintasan terpendek pada suatu game berbasis ubin (*Tile Based Game*).

ISSN : 2085-4218

Kecerdasan Buatan Untuk Game

Artificial Intelligence (AI) atau kecerdasan buatan merupakan suatu program komputer yang bertindak dan berpikir seperti manusia dan juga bertindak dan berpikir secara rasional pada saat yang bersamaan. AI banyak digunakan di berbagai bidang, dan salah satunya digunakan dalam sebuah game yang biasa disebut Game AI. Game Artificial Intelligence (Game AI) merupakan suatu kecerdasan buatan yang digunakan dalam game untuk membuat unit-unit yang ada pada game dapat mengambil keputusan yang cerdas ketika pada game tersedia banyak pilihan untuk situasi tersebut, dan menghasilkan perilaku yang relevan, efektif dan berguna. Sehingga membuat game tersebut menjadi lebih seimbang dan sesuai dengan gameplay yang dirancang.

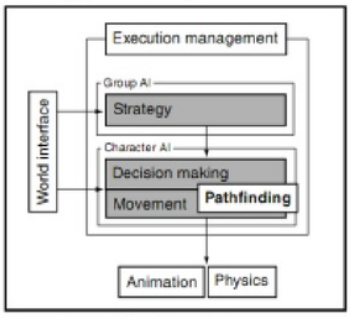


Gambar 1. Model game AI [1]

Game AI menggunakan teknik-teknik kecerdasan buatan namun dengan penerapan yang lebih sederhana dikarenakan keterbatasan komputasi dan kemampuan penyimpanan data pada game.

Pathfinding

Pathfinding merupakan salah satu jenis game AI. Dalam model game AI proses pencarian jalur terletak di perbatasan antara *decision making* dan *movement*, seperti ditunjukkan pada gambar 3. Mayoritas pathfinding dalam game menggunakan algoritma A*.



Gambar 3. Letak pathfinding dalam game AI model [1]

Algoritma A*

Algoritma A* didesain untuk pencarian jalur dari suatu titik ke titik lain. Menggunakan konsep graph dimana terdapat kumpulan node, yang merepresentasikan titik asal, tujuan, serta area untuk pencarian, dan edge, yang merepresentasikan jalan penghubung antar node. Proses pencarian jalur dimulai dengan menerima input berupa node asal dan node tujuan setelah itu dilakukan pencarian rute dengan algoritma A*. Output yang dihasilkan ada dua kemungkinan yaitu ada rute atau tidak ada rute dari node asal ke node tujuan.

Algoritma A*(AStar) merupakan algoritma *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best-First Search*. *Cost* yang diperhitungkan didapat dari *cost* sebenarnya ditambah dengan *cost* perkiraan. Dengan perhitungan *cost* seperti ini, algoritma A*(AStar) dapat optimal dan lengkap dalam menentukan jalur. Sama dengan algoritma dasar *Best First Search*, algoritma A*(AStar) ini juga menggunakan dua senarai: OPEN dan CLOSED. Terdapat tiga kondisi bagi setiap suksesor yang dibangkitkan, yaitu: sudah berada di OPEN, sudah berada di CLOSED, dan tidak berada di OPEN maupun CLOSED. Pada ketiga kondisi tersebut diberikan penanganan yang berbeda-beda [2]

ISSN : 2085-4218

Tile-Based Game

Tile-Based Game adalah *game* yang menggunakan *tile* sebagai salah satu elemen fundamental dalam permainan. *Tile* sendiri merupakan objek-objek kecil yang disatukan untuk membentuk area permainan pada *game*. Penggunaan *tile* tersebut dikarenakan untuk mengurangi kapasitas saat *game* dijalankan sehingga menjadi lebih ringan dan cepat dalam pemrosesan datanya.

Pada setiap *Tile-Based Game* terdapat *tileset* dan kumpulan *map*. *Tileset* adalah kumpulan dari *tiles* yang berbeda yang dibutuhkan untuk membuat sebuah *map game*. *Tileset* berupa satu dari setiap *tile* yang unik[3] . Salah satu contoh permainan berbasis ubin (*Tile Based Game*) ditunjukkan pada Gambar 3.

Tileset

Tileset adalah kumpulan *tile* yang digunakan untuk membangun *map game*. Kumpulan *tiles* tersebut dikombinasikan menjadi sebuah gambar yang besar. *Tileset* sering digunakan pada *game* 2D untuk membuat map yang kompleks dari *tiles* yang dapat digunakan kembali di dalam set. Ketika *tileset* dasar *map* ditampilkan, *tiles* disimpan didalamnya dan digunakan untuk menyusun kembali *map* untuk ditampilkan.

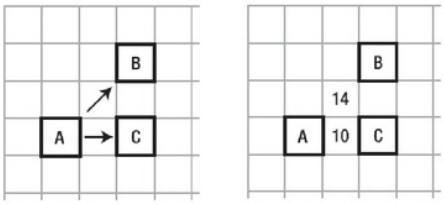
Gambar 3. *Tile-Based Game*

Perancangan Game

Algoritma A* Pada Game

Algoritma ini akan diterapkan pada kontrol gerakan objek *hero* pada *game*. Objek *hero* akan bergerak ke arah atau titik yang ditentukan menggunakan klik *mouse*, kemudian dengan menggunakan perhitungan dari Algoritma A* akan ditentukan jalur terpendek untuk menuju ke titik akhir tersebut.

Secara umum seperti digambarkan pada Gambar 4., pencarian jalur dari titik A menuju ke titik B membutuhkan waktu sekitar sepertiga lebih lama dari perjalanan ke titik C. Pada dasarnya dibutuhkan 1,41 kali lebih lama untuk melakukan perjalanan ke titik B dari titik A. Nilai 1,41 tersebut adalah *cost* perjalanan secara diagonal.



Gambar 4 Mencari jalur terdekat, A – B atau A – C

Di dalam dunia permainan berbasis ubin hanya terdapat dua pilihan gerakan, dan masing-masing memiliki *cost*, diagonal nilai *cost* 14, horizontal dan vertical nilai *cost* adalah 10.

Langkah-langkah pencarian jalur menggunakan Algoritma A* sebagai berikut :

1 Titik A adalah *parent node* dan merupakan langkah pertama dalam pencarian jalur.
2 Melakukan pemeriksaan terhadap delapan sel disekitar *parent node* untuk menentukan mana *node* berikutnya yang paling memungkinkan.
3 Delapan sel disekitar *parent node* diberikan *cost* acuan yang disebut sebagai G.

ISSN : 2085-4218

4 Selanjutnya mencari *node* disekitarnya yang lebih dekat dengan tujuan titik B, kemudian menjumlahkan *cost* perjalanan dari titik B ke setiap *node* di sekitarnya.
5 *Cost* setiap jalur heuristik juga sangat penting sehingga algoritma A*(AStar) mengacu pada *cost* ini sebagai H. *Cost* H pada setiap *node* disekitarnya dicari kemudian *cost* H dan G digabungkan sehingga menghasilkan *cost* F.
6 *Node* selanjutnya yang memiliki *cost* terendah adalah *node* yang tepat berada di sebelah kanan *parent node*. *Node* ini sekarang menjadi *parent node* baru yang potensial. Namun setelah dilakukan pengecekan kembali ternyata *node* tersebut bukanlah *node* terbaik, akan lebih baik jika bergerak secara diagonal naik dari titik A.
7 Melacak *parent node* terbaik untuk digunakan pada rute dengan menyimpan dua list:
   a) ***Closed List*** (daftar *node* yang tidak perlu diperiksa. Setiap kali *parent node* baru ditemukan dimasukkan ke dalam *closed list*)
   b) ***Open List*** (daftar *node* yang berada di sekitar *parent node* dan merupakan *node* yang perlu diperiksa)
8 Saat *parent node* potensial memeriksa *node* sekitarnya, terlihat bahwa setiap *node* G sebelumnya berada pada *open list*.
9 Untuk menemukan *cost* G yang baru, maka nilai G diambil dari nilai *parent node* saat ini dan ditambahkan dengan *cost* perjalanan satu *node*. Dan menghasilkan *cost* total baru G.
10 Selanjutnya dengan cara yang sama dicari *cost* G yang baru di setiap *node* sekitar. Ditemukan bahwa *cost* G dari setiap *node* tersebut ternyata lebih besar, sehingga *parent* tidak perlu dirubah.
11 Pencarian dari *node* yang akan di-tes berikutnya dengan mengabaikan dinding dan *parent node* sebelumnya. Kemudian dipilih *node* berikutnya dengan *cost* F terendah sebagai *parent node* baru seperti pada gambar 3.13 di bawah.
12 Jika beberapa *node* memiliki nilai yang sama, n*ode* atas dan bawah memiliki hasil nilai yang sama , akan dipilih *node* yang muncul pertama di *loop* yang berjalan pada pemeriksaan ini. Jika terjadi kesalahan, maka akan diperbaiki kemudian pada pemeriksaan selanjutnya. Dan pemeriksaan dilanjutkan.
13 Ulangi pemeriksaan hingga mencapai *node* tujuan, titik B. Saat seluruh jalur terbentuk lalu ditelusuri jalur dari *parent node* ke *parent node* untuk menghubungkan titik awal dan akhir bersamaan.
14 Pemeriksaan akan dihentikan ketika telah mencapai tujuan yaitu titik B, kemudian algoritma bekerja secara mundur dari titik B, berdasarkan *parent node* untuk membentuk jalur. Algoritma A*(AStar) menghasilkan *array* yang akan memberitahukan setiap *node* yang dibutuhkan untuk dilalui agar dapat menemukan jalur terpendek dari titik A ke titik B.

2. Pembahasan dan Pengujian

Dalam pengujian algoritma A*(AStar) ini terdapat tiga jenis heuristik yang berbeda yang akan diuji untuk digunakan dalam pencarian jalur. Ketiga jenis heuristik tersebut yaitu Manhattan, Euclidian, dan Diagonal. Tiap heuristik tersebut memiliki keunikan sendiri untuk menentukan jalur. Heuristik tersebut memiliki cost yang sama namun cara mencari rutenya yang memiliki perbedaan. Pengujian dilakukan dengan melakukan kontrol mouse klik ke suatu titik tujuan untuk melihat respon hasil jalur yang ditempuh oleh masing – masing heuristic dari titik awal dimana obyek berada pada saat itu. Hasil pencarian untuk ketiga jenis heuristic ditunjukkan pada Gambar 4. Jalur yang dilewati ditandai dengan marking berwarna hitam pada tile.

a). Manhattan (b) Euclidean (c). Diagonal

Gambar 4. Pengujian Heuristik

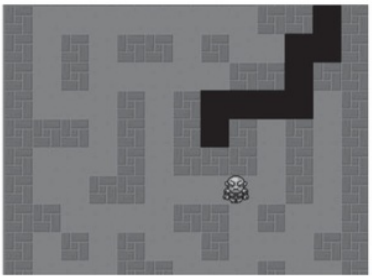
ISSN : 2085-4218

Seperti ditunjukkan pada Tabel 1. Metode Manhattan mengabaikan rute diagonal yang mungkin sehingga heuristik ini memiliki proses yang lebih cepat. Ini penting mengingat algoritma A*(AStar) sangat memakan proses CPU. Apabila akan digunakan untuk mencari jalur pada banyak karakter dalam sebuah *game* dapat menghemat waktu kinerja. Tetapi karena metode ini mengabaikan rute diagonal, tidak dapat menghasilkan jalur terpendek yang absolut. Rute yang diambil oleh metode Euclidean yaitu dengan berjalan lurus terlebih dahulu kemudian jalannya berbelok pada pertengahan jalur. Berbeda dengan metode Manhattan yang langsung berpindah jalur dari tile pertama dan bejalan lurus hingga titik tujuan. Sedangkan metode Diagonal jalur yang digunakan hampir mirip dengan Manhattan. Metode Euclidean memperhitungkan diagonal sehingga menghasilkan jalur yang sangat natural. Namun proses ini sedikit lebih lambat jika dibandingkan dengan metode Manhattan. Metode diagonal mengimbangi *cost* bergerak lurus atau diagonal. Sehingga menghasilkan perkiraan *cost* yang sangat akurat. Itu berarti A*(AStar) hanya perlu melakukan sedikit pencarian dan mendapatkan hasil yang lebih cepat, dan pasti menghasilkan jalur terpendek yang mungkin.

Tabel 1. Hasil Pengujian Algoritma A*

No	Jenis Heuristik	Waktu Proses	Penemuan Jalur Terpendek	Rute Yang Ditempuh
1	Manhattan	Cepat	Kurang Absolut	Langsung pindah jalur & lurus
2	Euclidean	Lambat	Absolut	Lurus, belok di pertengahan
3	Diagonal	Sangat Cepat	Absolut	Langsung pindah jalur & lurus

Hasil pencarian jalur dapat dilihat pada Gambar 5, perpindahan obyek dari titik awal ke titik tujuan ditunjukkan dengan marking warna hitam.



Gambar 5. Hasil Penemuan Jalur

3. Simpulan

Berdasarkan hasil pengujian algoritma A*(AStar) yang digunakan pada *game* ini teruji 100% efektif untuk menemukan jalur terpendek dari pergerakan objek dalam *game*. Pada pengujian heuristic, algoritma A*(AStar) menghasilkan jalur terpendek yang ditemukan adalah sama, namun rute yang di gunakan dari tiap heuristik berbeda.

4. Daftar Pustaka

[1] Roger E.Pedersen, 2003. *Game Design Foundation*. Wordware Publishing.
[2] Suyanto, 2011. *Artificial Intelligence Searching, Reasoning, Planning dan Learning*. Informatika, Bandung.
[3] Rex Van der Spuy, 2010. *Foundation Game Design with Flash*. Friendsofed, NewYork.

Penerapan Algoritma A* Untuk Pencarian Rute Terdekat Pada Permainan Berbasis Ubin (Tile Based Game)

ORIGINALITY REPORT

8%	8%	7%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	id.123dok.com Internet Source	3%
2	repository.usu.ac.id Internet Source	2%
3	pelita-informatika.com Internet Source	2%

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	On		